2.3. Specification of the Crystallographic Binary File (CBF/imgCIF)

BY H. J. BERNSTEIN AND A. P. HAMMERSLEY

2.3.1. Introduction

The Crystallographic Binary File (CBF) format is a complementary format to the Crystallographic Information File (CIF) (Hall *et al.*, 1991) supporting efficient storage of large quantities of experimental data in a self-describing binary format. The image-supporting Crystallographic Information File (imgCIF) is an extension to CIF to assist in ASCII debugging and archiving of CBF files and to allow for convenient and standardized inclusion of images, such as maps, diagrams and molecular drawings, into CIFs for publication. The binary CBF format is useful for handling large images within laboratories and for interchange among collaborating groups. For smaller blocks of binary data, either format should be suitable. The ASCII imgCIF format is appropriate for interchange of smaller images and for long-term archiving.

CBF is designed to support efficient storage of raw experimental data (images) from area detectors with no loss of information, unlike some existing formats intended for this purpose. The format enables very efficient reading and writing of raw data, and encourages economical use of disk space. It may be coded easily and is portable across platforms. It is also flexible and extensible so that new data structures can be added without affecting the present definitions.

These goals are achieved by a simple file format, combining a CIF-like file header with compressed binary information. The file header consists of ASCII text giving information about the binary data as CIF tag–value pairs and tables. Each binary image is presented as a text-field value, either as raw octets of binary data in a CBF data set, or as an ASCII-based encoding of the same binary information in a true ASCII imgCIF data set. The ASCII-based encoded format uses e-mail MIME (Multipurpose Internet Mail Extensions) conventions to encode the binary data (Freed & Borenstein, 1996*a*,*b*,*c*; Freed *et al.*, 1996; Moore, 1996). The present version of the format tries to deal only with simple Cartesian data. These are essentially the 'raw' diffraction data that typically are stored in commercial formats or individual formats internal to particular institutes. Other forms of binary image data could be accommodated. It is hoped that CBF will replace individual laboratory or institute formats for 'home-built' detector systems, will be used as an inter-program data-exchange format, and will be offered as an output choice by commercial detector manufacturers specializing in X-ray and other detector systems. In this chapter we discuss the basic framework within which binary data and images are stored. The categories and data items that are used to describe beam and equipment axes, rastering methodologies, and image compression techniques are described in Chapter 3.7. The CBF/imgCIF dictionary is given in Chapter 4.6. An application programming interface (API) for the manipulation of image data is described in Chapter 5.6.

Affiliations: HERBERT J. BERNSTEIN, Department of Mathematics and Computer Science, Kramer Science Center, Dowling College, Idle Hour Blvd, Oakdale, NY 11769, USA; ANDREW P. HAMMERSLEY, ESRF/EMBL Grenoble, 6 rue Jules Horowitz, France.

2.3.2. CBF and imgCIF

CBF and imgCIF are two aspects of the same format. Since CIFs are pure ASCII text files, it was necessary to define a separate binary format to allow the combination of pseudo-ASCII sections and binary data sections. In the binary-file CBF format, the ASCII sections conform closely to the CIF standard but must use operating-system-independent 'line separators'. In order to facilitate interchange of files, an API that writes CBF files should use `\r\n` (carriage return, line feed) for the line separator. Use of this line separator allows the ASCII sections to be viewed with standard system utilities (*e.g.* 'more', 'pg') on a very wide range of operating systems (*e.g.* Unix, MacOS and Windows). However, an API that reads CBF format must accept any of the following three alternative line terminators as the end of an ASCII line: `\r`, `\n` or `\r\n`. As for all CIF data sets, an imgCIF file conforms to the normal text file-writing conventions of the system on which it is written. imgCIF is also one of the two names of the CIF dictionary (see Chapter 4.6) that contains the terms specific to describing image data in both CBF and imgCIF data sets. Thus a CBF or imgCIF data set uses data names from the CBF/imgCIF dictionary and other CIF dictionaries.

The general structure of a CBF or imgCIF data set is shown in Example 2.3.2.1. After a special comment to identify the file type (a so-called 'magic number') and any other initial comments, the data set begins with a '**`data_`*`blockname`***' which gives the name of the data block. Tags and values that describe the image data and how they were collected come next. For efficiency in processing, it is recommended that all the descriptive tags come before the actual image data. This recommendation is a *requirement* for the binary CBF format. It is *optional* for the ASCII imgCIF format. The image data are given as the value of the tag **`_array_data.data`**. The image data are given in a text field, using MIME conventions to describe the encoding.

2.3.2.1. A simple example

Before describing the format in full, we start by showing a simple but important and complete use of the format: that of storing a single detector image in a file together with a small amount of auxiliary information. This is intended to be a useful example that can be understood without reference to the full definitions. It also serves as an introduction or overview of the format definition. This example uses CIF DDL2-based dictionary items (see Chapter 2.6).

Example 2.3.2.2 relates to an image of 768 × 512 pixels stored as 16-bit unsigned integers, in little-endian byte order (this is the native byte ordering on a PC). The pixel sizes are 100.5 × 99.5 μm.

The example will be presented and discussed in three sections. The circled numerals (*e.g.* ①) are included to allow us to comment on portions of the example. They are not part of the CBF/imgCIF format.

The line marked by ①, starting with a hash character (**`#`**), is a CIF and CBF comment line. As a first line, the pattern of three hashes followed by 'CBF' helps to identify the data set as a CBF. It is a so-called 'magic number'. The text **`###CBF: VERSION`** must be present as the very first line of every CBF file. Following

references